AD 741845

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)*

Informatics, Inc.
Systems and Services Company
6000 Executive Blfd., Rockville, Md. 20852

2a. REPORT SECURITY CLASSIFICATION

UNCLASSIFIED

2b. GROUP

3. REPORT TITLE

ANALYSIS OF AN INFORMATION SYSTEM MODEL WITH TRANSFER PENALTIES

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Scientific Interim

5. AUTHOR(S) *(First name, middle initial, last name)*

Thomas C. Lowe

6. REPORT DATE

12 April 1972

7a. TOTAL NO. OF PAGES

40

7b. NO. OF REFS

21

8a. CONTRACT OR GRANT NO.

F44620-71-C-0070

b. PROJECT NO. 9769

c. 61102F

d. 681304

9a. ORIGINATOR'S REPORT NUMBER(S)

9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*

AFOSR - TR - 72 - 0991

10. DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited.

11. SUPPLEMENTARY NOTES

TECH, OTHER

12. SPONSORING MILITARY ACTIVITY

Air Force Office of Scientific Research(NM)
1400 Wilson Boulevard
Arlington, Virginia 22209

13. ABSTRACT

A general model of systems consisting of "control units" and "passive units" is established. These units could be instructions and data in computer programs, for example, or information processing modules and files in data management systems; other applications are also suggested. The model contains both structual and behavioral information. The sets of units of the model are partitioned, and the resulting subsets correspond to different groups of elements of real systems.

Certain classes of communications between the subsets result in "boundary crossings," with which penalties are associated. An analysis of the model is performed, in order to determine the expected frequency of boundary crossing. Synthesis of the subsets in order to reduce the number of boundary crossings is discussed, under the presumption that the other parameters of the model remain fixed.

ANALYSIS OF AN INFORMATION SYSTEM MODEL WITH TRANSFER PENALTIES*

Thomas C. Lowe

12 April 1972

* This work was supported by the Air Force Office of Scientific Research (AFSC), United States Air Force, under contract F44620-71-C-0070.

informatics inc
Systems and Services Company
6000 Executive Boulevard
Rockville, Maryland 20852
(301) 770-3000 Telex: 89-521

Abstract

A general model of systems consisting of "control units" and "passive units" is established. These units could be instructions and data in computer programs, for example, or information processing modules and files in data management systems; other applications are also suggested. The model contains both structural and behavioral information. The sets of units of the model are partitioned, and the resulting subsets correspond to different groups of elements of real systems.

Certain classes of communications between the subsets result in "boundary crossings," with which penalties are associated. An analysis of the model is performed, in order to determine the expected frequency of boundary crossing. Synthesis of the subsets in order to reduce the number of boundary crossings is discussed, under the presumption that the other parameters of the model remain fixed.

Index Terms

information system model
data management
information retrieval
computer network
virtual memory
paged system
segmented program
partitioned system
boundary crossing

1. INTRODUCTION

This paper presents a class of hypothetical "partitioned systems", which exhibit many of the characteristics of several types of information systems. This similarity makes these systems useful as models in both analysis and design of real systems, Partitions define "boundaries" in partitioned systems; the transfer of information and control across such boundaries results in "boundary crossings". In real systems a cost is incurred with each boundary crossing occurrence in the model, so it is worthwhile to reduce the frequency of boundary crossing.

Following this introduction, partitioned systems are defined formally and illustrated. The fundamental results obtained are expressions for the expected number of boundary crossings, which are presented in the section on boundary crossing analysis. Problems of synthesizing partitions in order to minimize the expected number of boundary crossings are then discussed.

An informal description of the model is useful here. A system is a septuple whose elements are a set of control units, a set of passive units, and five arrays describing various characteristics of the units. At any instant in time, a control unit is either active or inactive. When active, a control unit may reference one or more passive units. A partitioned system is a system, with a partition of the set of control units, and a partition of the set of passive units. Each such partition is a collection of nonempty, disjoint subsets, whose union equals the set of units. Units which belong to the same subset are in the same equivalence class of the partition, and conversely.

A boundary crossing results when control unit activity passes between equivalence classes of the control unit partition. A boundary crossing also results when an active control unit references a passive unit, provided that no element of the equivalence class containing that passive unit was referenced by the previously active control unit.

Many types of information systems may be represented by partitioned system models. For example, control units can model programs which process the data base in data management and information retrieval systems, while passive units correspond to portions of the data base. Boundary crossings are associated with the accessing of different groups of data, and with program activations. Thus, organization of the data base and its processors for optimization of the system's performance is an application area for boundary crossing analysis. Similarly, the results are applicable to problems of information network organization. This includes both computer networks, and others such as library networks [1].

Technological advances in the development of memory devices have made large volumes of data available in mass storage. Complex addressing schemes are required in order to access data in such devices. In the partitioned system model, the partitions of the set of passive units correspond to segments of storage, and boundary crossing is related to the overhead involved in the accessing of data.

The work reported here evolved from investigation of several problems of paged memory systems. The first analytical models developed relied only on Boolean connectivity relationships between control units [2,3,4,5]. Later, probabilistic models were introduced and the problem of gathering required values of the parameters studied [6,7,8]. The generalization of those models lead to the present concept of partitioned systems.

It is assumed below that control unit activity exhibits the Markov property. This assumption has been discussed in connection with computer applications [9, 10, 11, 12] and other diverse areas such as circulation of library books [13]. The present model includes a control unit transition probability matrix and an entrance probability vector. These are used to derive the values of other fundamental parameters[1]; however, those values may sometimes be measured directly [6]. When such observation is possible, the Markov chain is unnecessary, as values which may be derived from it are available more readily from other sources.

2. PARTITIONED SYSTEMS

In order to define partitioned systems and their behavior, first the elements of systems exclusive of any partitions are presented. Following a discussion of the behavior of systems in the passage of time, partitions are defined. The present section ends with a discussion of control unit activity, thus completing preparation for the section on boundary crossing analysis.

A system is a septuple ($\mathfrak{A}$, $\mathfrak{B}$, $\underline{G}$, $\underline{H}$, $\underline{E}$, P, Q), where:

$\mathfrak{A}$ is a set of m control units: $\mathfrak{A}=\{\alpha_i;\ i=1, 2, \dots, m\}$;

$\mathfrak{B}$ is a set of n passive units: $\mathfrak{B}=\{\beta_u;\ u=1, 2, \dots, n\}$;

$\underline{G}$ is a vector of control unit volumes in which g_i is the value of the volume measure of α_i: $\underline{G} = [g_i;\ i=1, 2, \dots, m]$;

$\underline{H}$ is a vector of passive unit volumes in which h_u is the value of the volume measure of β_u: $\underline{H} = [h_u;\ u=1, 2, \dots, n]$;

$\underline{E}$ is a vector of control unit entrance probabilities: $\underline{E} = [e_i;\ i=1, 2, \dots, m]$;

P is a matrix of control unit transition probabilities: $P = [p_{ij};\ i=1, 2, \dots, m;\ j=1, 2, \dots, m]$;

Q is a matrix of passive unit references in which q_{iu} is the zero-one variable indicating a relationship[2] between α_i and β_u: $Q = [q_{iu};\ i=1, 2, \dots, m;\ u=1, 2, \dots, n]$.

Interpretation

A control unit is always in one of two states: active ɔr inactive. Of the m control units of a system, at most one can be active at a time. A system is also either active or inactive; it is active if and only if one of its control units is active.

In the continuing passage of time, discrete instants at which events of interest occur are designated $t_i, t_j, \ldots$. It is required that $i < j$ whenever t_i precedes t_j, and that $i = j$ only when $t_i = t_j$. The events of concern here are the activation of control units.

Consider a system that is inactive at some time t_0. At some later time t_1, control unit α_{i_1} becomes active. Therefore, by definition, the system also becomes active at time t_1. Control unit α_{i_1} remains active for some interval, until at time t_2 control unit α_{i_2} becomes active and α_{i_1} inactive. This continues for a finite time, until at some time t_{r+1}, control unit α_{i_r} becomes inactive and no other unit becomes active. Since no control unit is active, the system is inactive. The time interval from t_1 to t_r is called a system epoch.

For every system epoch, some control unit must be the first control unit to become active. The probability that α_i is active at the very start of an epoch is e_i, an element of the probability entrance vector $\underline{E}$.

During a system epoch, given that control unit α_i is active at time t_k, the probability that α_j is active at time t_{k+1} is p_{ij}, an element of the probability transition matrix P. It is required that $\lim_{k \to \infty} P^k = 0$, and that P not be a function of time.

Whenever a control unit is active, it references a fixed set of zero or more passive units. Given that control unit α_i is active at time t_k, the passive units referenced at that time are those β_u for which $q_{iu}=1$, where q_{iu} is an element of the zero-one passive unit reference matrix Q.

Certain constraints exist on allowable partitions of the sets $\mathfrak{A}$ and $\mathfrak{B}$. These are related to the volumes of the individual control and passive units, which in turn may be associated with the information content of those units. The volumes of α_i and β_u are, respectively, the positive real numbers g_i and h_u.

Partitioned Systems

When the sets $\mathfrak{A}$ and $\mathfrak{B}$ are partitioned, in the usual mathematical sense, the system is said to be partitioned. There exist sets $A_1, \ldots, A_k, \ldots, A_K$ and $B_1, \ldots, B_w, \ldots, B_W$ such that[9]:

$$A_k \neq \Phi \text{ and } B_w \neq \Phi; \text{ and}$$
$$k \neq k' \Rightarrow A_k \cap A_{k'} = \Phi \text{ and } w \neq w' \Rightarrow B_w \cap B_{w'} = \Phi; \text{ and}$$
$$\bigcup_{k=1}^{K} A_k = \mathfrak{A} \text{ and } \bigcup_{w=1}^{W} B_w = \mathfrak{B}.$$

For the present purposes, sufficient information on the partitioning of a system can be contained in two zero-one matrices. The $m \times m$ matrix R consists of elements

$$r_{ij} = 1 \Leftrightarrow (\exists A_k)(\alpha_i, \alpha_j \in A_k), \tag{1}$$

and the $n \times n$ matrix S has elements

$$s_{uv} = 1 \Leftrightarrow (\exists B_w)(\beta_u, \beta_v \in B_w). \tag{2}$$

The matrices R and S thus describe the equivalence classes of the partitions; they are a convenient form for the following development.

CONTROL UNIT ACTIVITY

The fundamental parameters used to describe the control portion of a system are discussed in this section. The use of transition probabilities allows the behavior of control units to be described by an absorbing Markov chain[4]. The model also includes a vector specifying the probability that each control unit is the initially active unit in a system epoch.

Transition Probabilities

The probability that control unit α_j becomes active when unit α_i becomes inactive is p_{ij}. The condition of system inactivity can be represented by a single unit α_{n+1}, located outside of the system, such that $p_{n+1,n+1} = 1$. Then,

$$\begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1n} & p_{1,n+1} \\ p_{21} & p_{22} & \cdots & p_{2n} & p_{2,n+1} \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ p_{n1} & p_{n2} & \cdots & p_{nn} & p_{n,n+1} \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \qquad (3)$$

forms an absorbing Markov chain with a single ergodic state. For any α_i, the probability that the system will become inactive at the time that α_i becomes inactive, is $p_{i,n+1}$. Since

$$p_{i,n+1} = 1 - \sum_{j=1}^{n} p_{ij}\,, \qquad (4)$$

and the bottom row of (3) is always $[0\ 0\ \ldots\ 0\ 1]$, that row and the rightmost column contain redundant information and may be eliminated. The control unit transition probability matrix thus is

$$P = \begin{bmatrix} p_{11} \cdots p_{1m} \\ \vdots \quad \cdots \quad \vdots \\ p_{m1} \cdots p_{mm} \end{bmatrix} . \tag{5}$$

Whenever system activity is initiated, there must be some control unit that is the first to be active. It is allowed that different units be initially active, in successive system epochs. For each α_i, the probability that α_i is the first active control unit at the start of a system epoch is e_i. It is required that

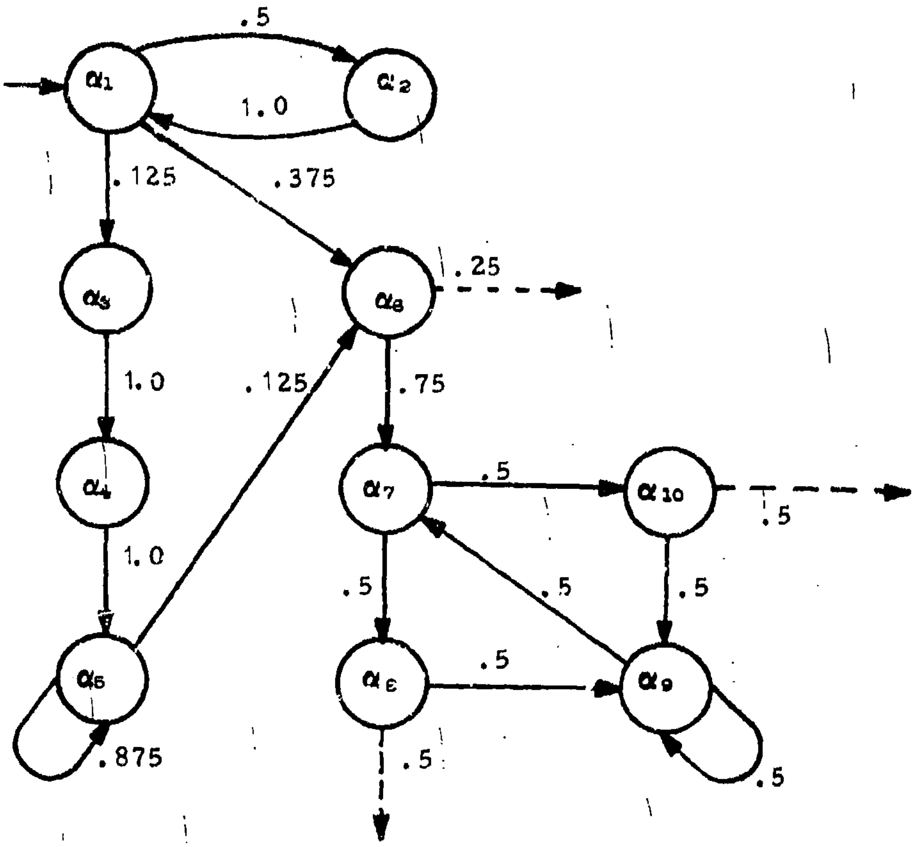
$$\sum_{i=1}^{m} e_i = 1 . \tag{6}$$

In the event that only one control unit is eligible to be the first one active, $\underline{E} = [e_1, e_2, \ldots, e_m]$ contains $m-1$ zeros and a single one.

Example

Figure 1 shows an example set of control units and the associated matrix P, represented on a directed graph. Transition probabilities are shown on the edges; the system can become inactive following a period of control in units α_6, α_8, or α_{10}. The probability of control exiting the system from these units is depicted by dotted lines in the graph. In the matrix P, the probability of control leaving the system from a control unit can be obtained by subtracting the sum of the elements of a row from unity. Thus, since the elements on the sixth row add to 0.75, the probability that the system will cease to be active following control in α_6 is 0.25.

In this example, control always is initiated in α_1. Therefore only e_1 is nonzero; it is equal to one, and all other elements of $\underline{E}$ are zero.



$$P = \begin{bmatrix} 0 & 0.5 & 0.125 & 0 & 0 & 0.375 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.875 & 0.125 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.75 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0.5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 \end{bmatrix}$$

$$\underline{E} = [1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]$$

Figure 1. . Illustration of Probability Transition Matrix P and Probability Entrance Vector E.

Control Flow

During a system epoch, control is transferred among the control units of a system. Certain characteristics of control flow important to the boundary crossing probme, are developed below. Many results follow immediately from an application of the elements of finite Markov chain theory. The general literature, Feller [14] and Kemeny and Snell [15] for example, give proofs omitted here.

Since P describes the transient states of an absorbing Markov chain,

$$\lim_{k \to \infty} P^k = 0, \tag{7}$$

and

$$F = \sum_{k=0}^{\infty} P^k \tag{8}$$

is bounded. Indeed, equation (7) is necessary and sufficient for the inverse of (I-P) to exist and to be equal to F. Also, given that control unit α_i is active at the beginning of a system epoch, the expected number of times that α_j is active during that epoch is f_{ij}.

An important question is the following: During a system epoch, how many times is control unit α_j expected to become active? It is active $e_i f_{ij}$ times owing to α_i being initially active. Summing over all possible initial active units, the number of times α_j is expected to become active is

$$\gamma_j = \sum_{i=1}^{s} e_i f_{ij} . \tag{9}$$

Thus, the vector $\underline{\Gamma} = [\gamma_1, \gamma_2, \ldots, \gamma_s]$ gives the expected number of times each control unit will become active during a system epoch. It is known that $F = (I-P)^{-1}$, so from equation (9) it follows that

$$\underline{\Gamma} = \underline{E}(I-P)^{-1}. \qquad (10)$$

Given control units α_i and α_j, an important quantity is the expected number of times that control passes directly from α_i to α_j during a system epoch. Given that unit α_i is active, the probability that α_j will become active next is p_{ij}. Since the total expected number of times that α_i is activ is γ_i, the expected number of control transfers[6] from α_i to α_j is

$$\tau_{ij} = \gamma_i p_{ij}. \qquad (11)$$

Example

For the set of control units depicted in Figure 1, the fundamental matrix is

$$(I-P)^{-1} = \begin{bmatrix} 2 & 1 & 0.25 & 0.25 & 2 & 1 & 1.5 & 0.75 & 1.5 & 0.75 \\ 2 & 2 & 0.25 & 0.25 & 2 & 1 & 1.5 & 0.75 & 1.5 & 0.75 \\ 0 & 0 & 1 & 1 & 8 & 1 & 1.5 & 0.75 & 1.5 & 0.75 \\ 0 & 0 & 0 & 1 & 8 & 1 & 1.5 & 0.75 & 1.5 & 0.75 \\ 0 & 0 & 0 & 0 & 8 & 1 & 1.5 & 0.75 & 1.5 & 0.75 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1.5 & 0.75 & 1.5 & 0.75 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1.5 & 2 & 0.5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0.5 & 2 & 1.5 \end{bmatrix}. \qquad (12)$$

Since $\underline{E}$ has only one nonzero element $(e_1 = 1)$, the vector $\underline{\Gamma}$ is identical to the first row of the matrix above (12). During a system epoch it is expected that α_1 will be active twice and α_2 once, as $\gamma_1 = [(I-P)^{-1}]_{11} = 2$ and $\gamma_2 = [(I-P)^{-1}]_{12} = 1$.

The matrix of elements τ_{ij} giving the expected number of transfers between units may be obtained from equations (10) and (11):

$$\begin{bmatrix} 0 & 1 & .25 & 0 & 0 & .75 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & .25 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & .25 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.75 & .25 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & .75 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & .75 & 0 & .75 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .375 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & .75 & 0 & .75 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .375 & 0 \end{bmatrix} . \quad (13)$$

Figure 2 illustrates these results. The value adjacent to each edge shows the expected number of times that the transition of control depicted by that edge will occur during a system epoch. For example, it can be seen from inspection of Figure 1 that when control leaves the α_1-α_2 pair of control units, the probability that it enters α_3 is $p_{13}/(1-p_{12}) = 1/4$. As control leaves the α_1-α_2 pair only once during a period of system activity, the expected number of times that control will pass from α_1 to α_3 is also 1/4. This is also the expected number of times control passes from α_4 to α_5. If in α_5, control is expected to loop through that unit $(7/8) + (7/8)^2 + (7/8)^3 + \ldots = 7$ times. Given that control is expected to enter α_5 1/4 time, the number of times that the edge connecting α_5 with itself will be traversed is $7/4 = 1.75$.

3. BOUNDARY CROSSING ANALYSIS

A definition of boundary crossing is presented below. Given the matrix Q, the sets A_k and B_v, and the history of control flow during a specific system epoch, the definition may be applied to determine the number of boundary crossings which occurred during that epoch.

When given the matrix P and vector $\underline{E}$, the control flow analysis of the preceding section may be applied in order to determine the expected number of boundary crossings for a system epoch.

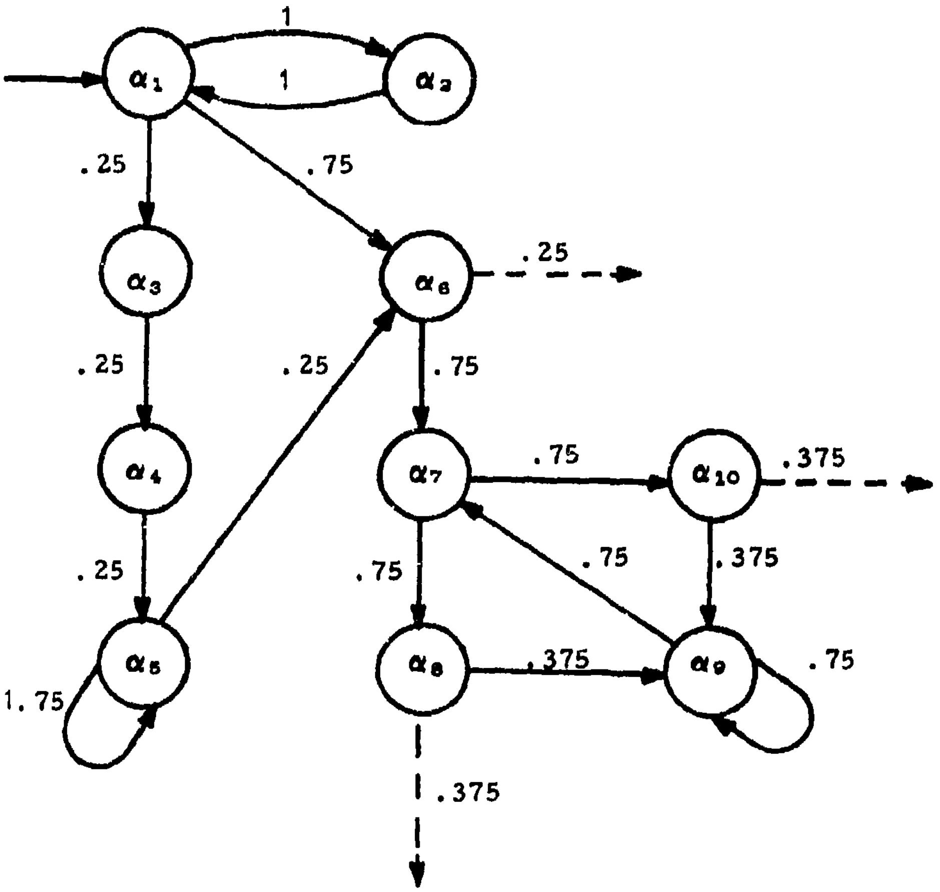


Figure 2. Expected Number of Control Transition Between Units During a System Epoch

Boundary crossings arise from transfers of control. Assume that α_j is active immediately before time t_0, and that α_i becomes active at time t_0. Associated with this transition is a boundary crossing if α_i and α_j are elements of different sets A_k, $A_{k'}$. A boundary crossing also results for each set B_w both containing some elements referenced by α_j and containing no elements referenced by α_i. When a system first becomes active, boundary crossings result from the need for an A_k containing the first active control unit and those B_w containing passive units which it references.

DEFINITION OF BOUNDARY CROSSING

Two functions, $\Delta(c)$ and $\sigma(\alpha_i)$, are useful. The former describes the history of control flow in a system during a specific system epoch; it records a particular history of control flow. The latter is concerned with describing one aspect of a partitioned system, indicating those sets B_w containing passive units β_u referenced by the control unit α_i.

For a given system epoch, the control units become active in some sequence: $\alpha_{i_1}, \alpha_{i_2}, \ldots, \alpha_{i_c}, \ldots, \alpha_{i_f}$. Considering one such specific system epoch, the function generating the sequence is:

$$\Delta(c) = \alpha_{i_c}, \quad \text{for } c = 1, \ldots, f. \qquad (14)$$

Given any α_i, it is useful to be able to refer to the set of sets B_w containing β_u referenced by that α_i:

$$\sigma(\alpha_i) = \{B_w \mid \exists \beta_u (q_{iu} = 1 \wedge \beta_u \in B_w)\}, \quad \text{for } \alpha_i \in \mathfrak{A}. \qquad (15)$$

Boundary Crossing: Control Units

A single boundary crossing results from the initial activation of a control unit, the first such unit to become active during a system epoch. Subsequently, an additional boundary crossing results each time control transfers between control units that are elements of different subsets A_k of $\mathfrak{A}$.

Thus, a boundary crossing occurs whenever some value of c satisfies one of the expressions below:

$$c = 1; \tag{16}$$

$$[1 < c \leq f] \wedge [\exists A_k, A_{k'} (\Delta(c-1) \in A_k \wedge \Delta(c) \in A_{k'} \wedge A_k \neq A_{k'})]. \tag{17}$$

Boundary Crossing: Passive Units

Multiple boundary crossings may result indirectly from the initial activation of a control unit, the first such unit to become active during a system epoch. The number of boundary crossings so generated is equal to the number of distinct subsets B_w of $\mathfrak{B}$ containing one or more passive units referenced by that control unit. Subsequently, an additional boundary crossing occurs each time an active control unit references one or more passive units which are elements of some B_w, provided that B_w contains no passive units referenced by the previously active control unit.

The number of boundary crossings so produced is

$$|\sigma(\Delta(1))|, \text{ for } c = 1; \tag{18}$$

and

$$|\sigma(\Delta(c)) \cap \overline{\sigma(\Delta(c-1))}|, \text{ for } c = 2, \ldots, f. \tag{19}$$

Example: Control Units

Figure 3 shows the same set of control units illustrated in Figure 2, but partitioning of the set is also indicated[6]. Suppose that, during a system epoch, the sequence of active units is $\alpha_1, \alpha_2, \alpha_1, \alpha_6, \alpha_7, \alpha_{10}$. One boundary crossing occurs at c=1, according to expression (16). At c=2, $\Delta(c-1) = \alpha_1$ and $\Delta(c) = \alpha_2$; both of these control units are elements of A_1, so expression (17) is not satisfied. The table below summarizes the boundary crossings.

c	Δ(c)	Δ (c-1)	A_k containing Δ(c)	$A_{k'}$ containing Δ (c-1)	Expression satisfied	Boundary crossings
1	α_1	-	A_1	-	16	1
2	α_2	α_1	A_1	A_1	-	0
3	α_1	α_2	A_1	A_1	-	0
4	α_6	α_1	A_3	A_1	17	1
5	α_7	α_6	A_4	A_3	17	1
6	α_{10}	α_7	A_9	A_4	17	1
					Total	4

Thus, for this particular system and system epoch, four boundary crossings arise from the partition of $\mathfrak{A}$.

Example: Passive Units

Figure 4 shows a system including passive units[6]. Squares are used for passive units, and circles for control units. Directed edges represent control flow; edges without arrowheads indicate the referencing of passive units by control units. Assume that the sequence of control units activity is: $\alpha_1, \alpha_4, \alpha_2, \alpha_3$. Considering only the boundary crossings resulting from passive units, the table below shows that this particular system epoch results in eight boundary crossings.

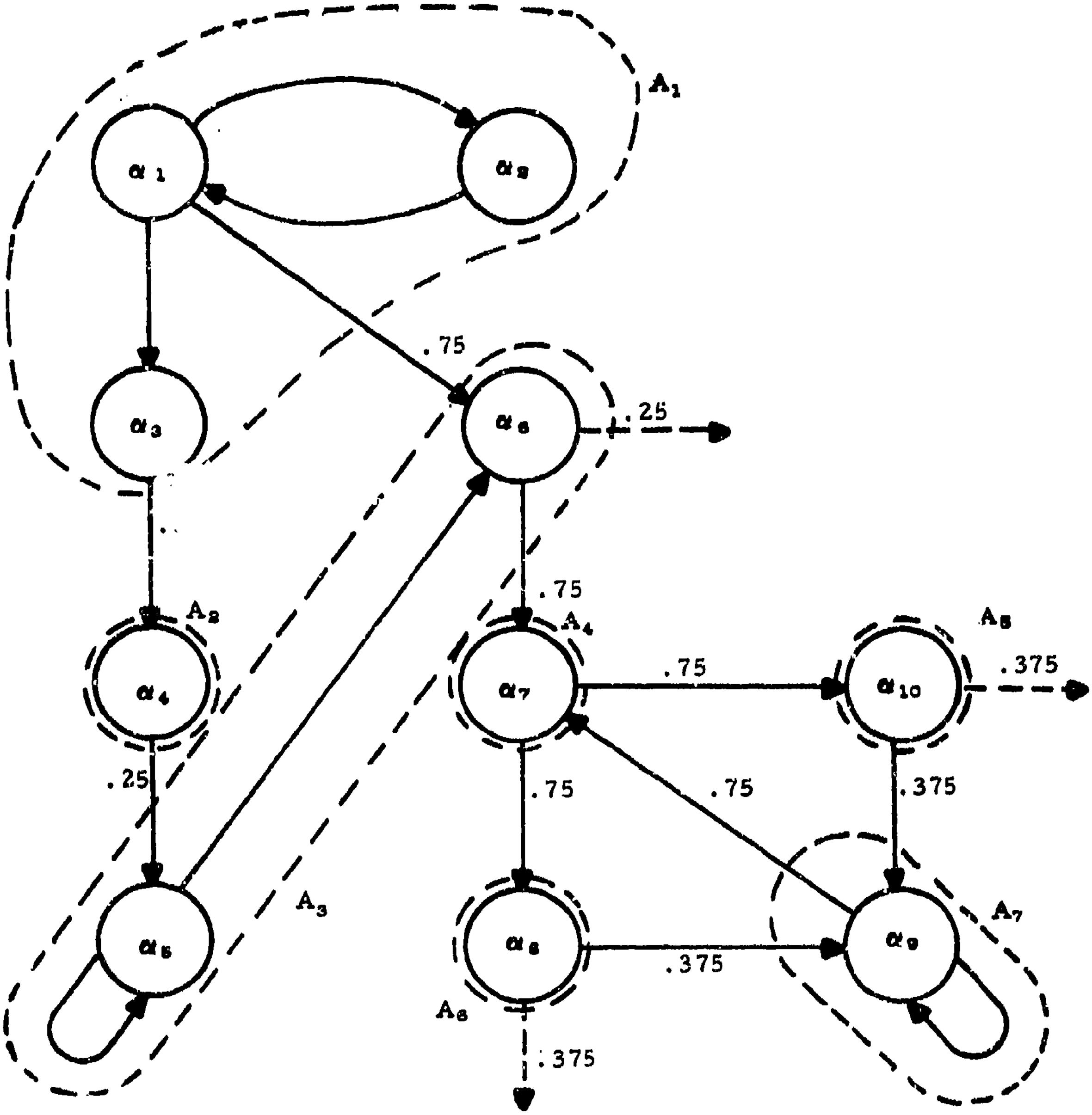


Figure 3. System Showing Partition of $\mathfrak{U}$.

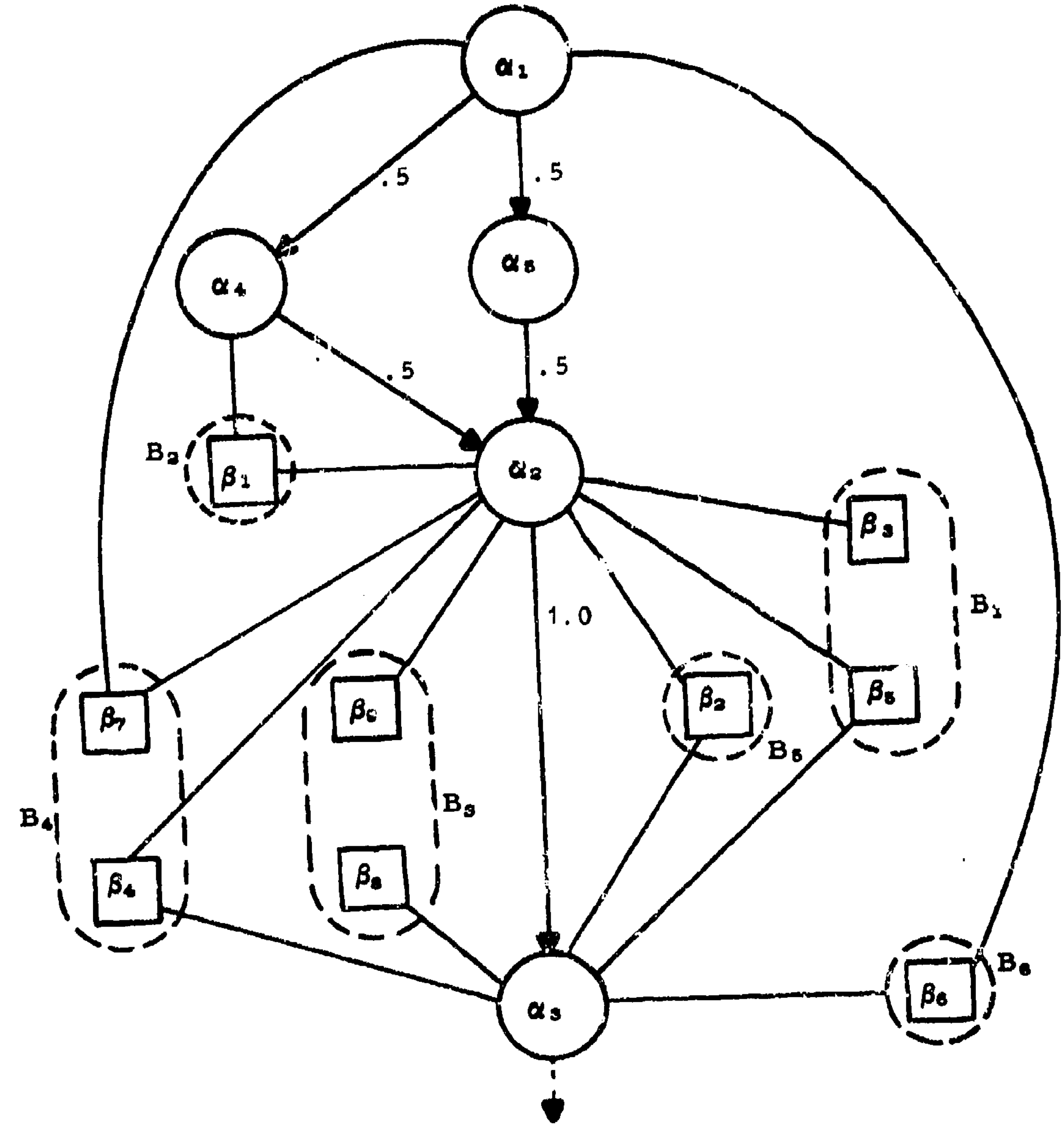


Figure 4. System Showing Partition of $\mathcal{B}$.

c	$\Delta(c)$	$\Delta(c-1))$	$(\Delta(c))$	$(\Delta(c-1))$	$\sigma(\Delta(c))\cap\overline{\sigma(\Delta(c-1))}$	Expression Satisfied	Number of Boundary Crossings
1	α_1	-	$\{B_4, B_6\}$	-	-	18	2
2	α_4	α_1	$\{B_8\}$	$\{B_4, B_6\}$	$\{B_8\}$	19	1
3	α_6	α_4	$\{B_1, B_2, B_3, B_4, B_8\}$	$\{B_8\}$	$\{B_1, B_2, B_4, B_3\}$	19	4
4	α_8	α_6	$\{B_1, B_2, B_4, B_8, B_6\}$	$\{B_1, B_2, B_3, B_4, B_8\}$	$\{B_6\}$	19	1
						Total	8

Expected Number of Boundary Crossings

Given the partitioning of $\mathfrak{A}$ into sets A_k and of $\mathfrak{B}$ into sets B_v, and the passive unit reference matrix Q, a knowledge of control flow for any particular system epoch is all that is required in order to count the boundary crossings occurring during that epoch. The next problem is to develop a method of estimating the number of boundary crossings a priori.

The expected number of times that some c satisfies the conditions that $\Delta(c) = \alpha_j$ and $\Delta(c-1) = \alpha_i$ is τ_{ij}, from equation (11). Equation (10) relates this quantity to $\underline{E}$ and P. The expected number of boundary crossings for a given partitioned system can therefore be determined by an application of control flow analysis to the definition of boundary crossing. A formal development follows two illustrative examples.

Example: Control Units

The numerical values associated with certain edges in Figure 2 indicate the expected number of transitions during a system epoch. They are shown only for edges joining control units in different sets A_k. Thus, the edge from α_8 to α_7 means that during a system epoch the expected number of times that there will exist some c such that $\Delta(c-1) = \alpha_8$ and $\Delta(c) = \alpha_7$ is 0.75. Expression (13) shows that $\tau_{87} = 0.75$.

The pairs of control units which meet necessary conditions of boundary crossing imposed by condition (17) are those pairs (α_i, α_j) such that α_i and α_j are elements of different subsets A_k, and for which $p_{ij} > 0$. In this example they are (α_1, α_6), (α_3, α_4), (α_4, α_5), (α_6, α_7), (α_7, α_8), (α_7, α_{10}), (α_8, α_9), (α_9, α_7), (α_{10}, α_9). Summing the expected number of times that these transitions will occur during a system epoch yields: 0.750 + 0.250 + 0.250 + 0.750 + 0.750 + 0.750 + 0.375 + 0.750 + 0.375 = 5.000. In addition, some control unit must become active at the beginning of a system epoch, satisfying condition (16) and contributing an additional boundary crossing. Therefore, considering only active units, the expected number of boundary crossings for this system during a system epoch is six.

Example: Passive Units

The numerical values associated with edges joining pairs of control units in Figure 3 are again values of τ_{ij}. E = [1 0 0 0 0], so α_1 is always initially active. The table below summarizes computation of the expected number of boundary crossings, for passive units only. Each non-zero value of $|\sigma(\Delta(c)) \cap \overline{\sigma(\Delta(c-1))}|$ is multiplied by the expected number of times $\Delta(c)$ will follow $\Delta(c-1)$ in the sequence of active control units. To the sum is added $|\sigma(\Delta(1))| = |\sigma(\alpha_1)|$. Thus, the contributions from expressions (17) and (18) are totalled, weighted by the expected number of times those contributions will be included in a system epoch.

$\Delta(c-1)$ $(=\alpha_i)$	$\Delta(c)$ $(=\alpha_j)$	τ_{ij}	$\lvert\sigma(\Delta(c)) \cap \overline{\sigma(\Delta(c-1))}\rvert$	Expected Number of Boundary Crossings
α_1	α_4	0.5	1	0.5
α_1	α_5	0.5	0	0.0
α_4	α_2	0.5	4	2.0
α_5	α_2	0.5	5	2.5
α_2	α_3	1.0	1	1.0
Note: $\Delta(1)$ is always α_1.			$\lvert\sigma(\alpha_1)\rvert = 2$	2.0
			Total	8.0

Boundary Crossing Owing to Partition of $\mathfrak{A}$

The number of boundary crossings involving the sets of control units A_k is unity (from expression (16)) plus the number of times that expression (17) is satisfied for $c=2, \ldots, f$. The expected number of direct control transfers from α_i to α_j during a system epoch is given by expression (11); such a transfer corresponds to the case that $1<c\leq f$, $\Delta(c) = \alpha_j$ and $\Delta(c-1) = \alpha_i$. Expression (17) is satisfied whenever a transfer occurs for which the above is true, and $\alpha_i \epsilon A_k$, $\alpha_j \epsilon A_{k'}$, and $k' \neq k$. The latter condition is, from expression (1), equivalent to the requirement that r_{ij} be zero. The total expected number of boundary crossings resulting from the partitioning of $\mathfrak{A}$ is therefore obtained by summing over all possible pairs α_i, α_j:

$$Y = 1 + \sum_{i=1}^{m} \sum_{j=1}^{m} \tau_{ij}(1-r_{ij}). \tag{20}$$

Boundary Crossing Owing to Partition of $\mathfrak{B}$

The number of boundary crossings involving sets of passive units B_w is the value of expression (18), plus the sum of the values of (19), taken over $c=2, \ldots, f$:

$$|\sigma(\Delta(1))| + \sum_{c=2}^{f} |\sigma(\Delta(c)) \cap \overline{\sigma(\Delta(c-1))}|. \tag{21}$$

The expected number of boundary crossings owing to the passive units, Z, can be determined by including the probabilities of events leading to boundary crossing in expression (21) and summing over all possible events:

$$Z = \sum_{j=1}^{m} \Pr[\Delta(1) = \alpha_j] |\sigma(\alpha_j)| + \sum_{c=2}^{f} \sum_{i=1}^{m} \sum_{j=1}^{m} \Pr[\Delta(c) = \alpha_j \wedge \Delta(c-1) = \alpha_i] |\sigma(\alpha_j) \cap \overline{\sigma(\alpha_i)}|. \tag{22}$$

The problem of expressing Z in terms specifying the system and its partitioning is attacked as follows. For each α_j, the probability that $\Delta(1) = \alpha_j$ is found. For a given $c > 1$ and pair (α_i, α_j), the probability that $\Delta(c) = \alpha_j$ and $\Delta(c-1) = \alpha_i$ is also determined. These are applied to equation (22), along with closed form expressions for $|\sigma(\alpha_j)|$ and $|\sigma(\alpha_j) \cap \overline{\sigma(\alpha_i)}|$, in order to determine the expected value of Z. First, an expression is developed for $|\sigma(\alpha_j)|$.

Whenever α_j is active, it references some fixed set of passive units; the number of passive units referenced is

$$\sum_{u=1}^{n} q_{ju} , \qquad (23)$$

The number of distinct sets B_w involved must be counted, in order to determine $|\sigma(\alpha_j)|$. The function of j and u

$$[1 - \min(1, \sum_{v=1}^{u-1} q_{jv} s_{uv})] \qquad (24)$$

has value zero if there exist a B_w and β_v such that both β_u and β_v are elements of B_w and are referenced by α_j, and $v < u$; the value is one otherwise[7]. If (24) is multiplied by q_{ju} and summed over $u = 1, \ldots, n$, one term of unity is added to the sum for each B_w containing at least one element referenced by α_j. The term added corresponds to the β_u possessing the smallest subscript u, which β_u is an element of B_w referenced by α_j. Thus, each B_w containing any β_u referenced by α_j is counted once, and

$$|\sigma(\alpha_j)| = \sum_{u=1}^{n} q_{ju}[1 - \min(1, \sum_{v=1}^{u-1} q_{jv} s_{uv})]. \qquad (25)$$

The second term in equation (22) is evaluated by first noting that

$$|\sigma(\alpha_j) \cap \overline{\sigma(\alpha_i)}| = |\sigma(\alpha_j)| - |\sigma(\alpha_j) \cap \sigma(\alpha_i)|. \qquad (26)$$

In order to obtain $|\sigma(\alpha_j) \cap \sigma(\alpha_i)|$, equation (25) is modified to include a factor that is unity whenever the β_u contributing to the sum in (25) is an element of $\sigma(\alpha_i)$, and zero otherwise:

$$|\sigma(\alpha_j) \cap \sigma(\alpha_i)| = \sum_{u=1}^{n} q_{ju} [1 - \min(1, \sum_{v=1}^{u-1} q_{jv} s_{uv})] \cdot [\min(1, \sum_{v=1}^{n} q_{iv} s_{uv})]. \qquad (27)$$

Combining the above three equations yields

$$|\sigma(\alpha_j) \cap \overline{\sigma(\alpha_i)}| = \sum_{u=1}^{n} q_{ju} [1 - \min(1, \sum_{v=1}^{u-1} q_{jv} s_{uv})] \cdot [1 - \min(\sum_{u=1}^{n} q_{iv} s_{uv})]. \qquad (28)$$

The probabilities mentioned above are simply determined. By definition of $\underline{E}$, the probability that α_j is the first unit to become active during a system epoch is

$$\Pr[\Lambda(1) = \alpha_j] = e_j . \qquad (29)$$

For the present purposes, this is better stated in terms of P and $\underline{\Gamma}$. From equation (10),

$$\underline{E} = \underline{\Gamma}(I - P), \qquad (30)$$

so application of (11) yields

$$e_j = \gamma_j - \sum_{i=1}^{n} \tau_{ij} . \qquad (31)$$

During the f-1 transitions of control occurring when c=2, ... ,f, a total of τ_{ij} transitions from α_i to α_j are expected to occur. For a random value of c in that sequence, then,

$$\Pr[\Delta(c) = \alpha_j \wedge \Delta(c-1) = \alpha_i] = \tau_{ij}/(f-1). \tag{32}$$

Substitution of equations (25, 28, 29, 30, 31) into (22) yields:

$$\begin{aligned} Z = {} & \sum_{j=1}^{m} [\gamma_j - \sum_{i=1}^{m} \tau_{ij}][\sum_{u=1}^{n} q_{ju}[1-\min(1, \sum_{v=1}^{u-1} q_{jv} s_{uv})]] \\ & + \sum_{i=1}^{m} \sum_{j=1}^{m} \tau_{ij} \sum_{u=1}^{n} q_{ju}[1-\min(1, \sum_{v=1}^{u-1} q_{jv} s_{uv})] \\ & \cdot [1-\min(1, \sum_{v=1}^{n} q_{iv} s_{uv})]. \end{aligned} \tag{33}$$

Upon further algebraic reduction,

$$\begin{aligned} Z = {} & \sum_{j=1}^{m} \sum_{u=1}^{n} q_{ju}[1-\min(1, \sum_{v=1}^{u-1} q_{jv} s_{uv})][\gamma_j - \\ & \sum_{i=1}^{m} \tau_{ij} \cdot \min(1, \sum_{v=1}^{n} q_{iv} s_{uv})] \end{aligned} \tag{34}$$

is obtained. This form has intuitive appeal, since the product including γ_j counts the number of boundary crossings resulting from an activity in α_j. The second product counts the number of times α_j becomes active, but excludes those cases in which the preceding active unit references the same β_u as that contributing to the boundary crossing associated with α_j.

4. SYNTHESIS TECHNIQUES

Given a partitioned system, the preceding analysis can be used in order to determine the number of boundary crossings expected during a system epoch. Suppose that a system is given, with no partitions of $\mathfrak{A}$ and $\mathfrak{B}$. Presuming that partitioning is required, how may the subsets be specified in such a way as to determine the number of boundary crossings?

Depending on the size of the system, degree of connectivity and similar factors, different methods of partition selection are indicated. This paper presents only a brief discussion of the concepts involved and certain promising methods. Research into this problem is continuing, and a paper concerned with one facet of the problem is presently in review [16].

CONSTRAINTS

If no limitations were imposed on the formation of partitions, then the following trivial solution is possible: $A_1 = \mathfrak{A}$, $B_1 = \mathfrak{B}$, and so $Y = 1$ and $Z = 0$. The primary constraint chosen here is that, for each subset A_k, the sum of the volumes of all control units contained in A_k cannot exceed some constant (μ_c). A similar constraint condition is imposed on the partition of $\mathfrak{B}$:

for $i = 1, \ldots, m$,

$$\sum_{j=1}^{s} g_j r_{ij} \leq \mu_c , \tag{35}$$

and for $u = 1, \ldots, n$,

$$\sum_{v=1}^{n} h_v s_{uv} \leq \mu_p . \tag{36}$$

Since R and S describe equivalence classes of the partitions, they must be internally consistent. This requirement is satisfied if the following conditions are met:

$$r_{ii} = 1, \qquad (37) \qquad s_{uu} = 1, \qquad (38)$$

$$r_{ij} = r_{ji}, \qquad (39) \qquad s_{uv} = s_{vu}, \qquad (40)$$

$$r_{ij} = 1 \wedge r_{jk} = 1 \Rightarrow r_{ki} = 1, \qquad (41)$$

$$s_{uv} = 1 \wedge s_{vw} = 1 \Rightarrow s_{wu} = 1. \qquad (42)$$

In certain cases it is useful to replace constraints (41) and (42) by

$$r_{ij} + r_{jk} + r_{ki} \neq 2, \qquad (43)$$

and

$$s_{uv} + s_{vw} + s_{wu} \neq 2. \qquad (44)$$

In some methods of partition synthesis, constraints (37-44) must be explicitly applied. The formulation of other methods includes an explicit statement of certain constraints.

PARTITIONS OF $\mathfrak{A}$

The problem of partitioning $\mathfrak{A}$ is simpler than that of partitioning $\mathfrak{B}$; a few promising techniques are described below. The primary source of difficulty in applying these techniques arises from the large number of distinct partitions possible for even a small set $\mathfrak{A}$, as dramatically illustrated in Figure 5. There is no known general practical technique for partitioning $\mathfrak{A}$ for arbitrarily large systems when absolute minimization of the number of boundary crossings is required.

In addition to the brief observations below, it is useful to note that several techniques for solution of related problems have been compared by Coleman [17].

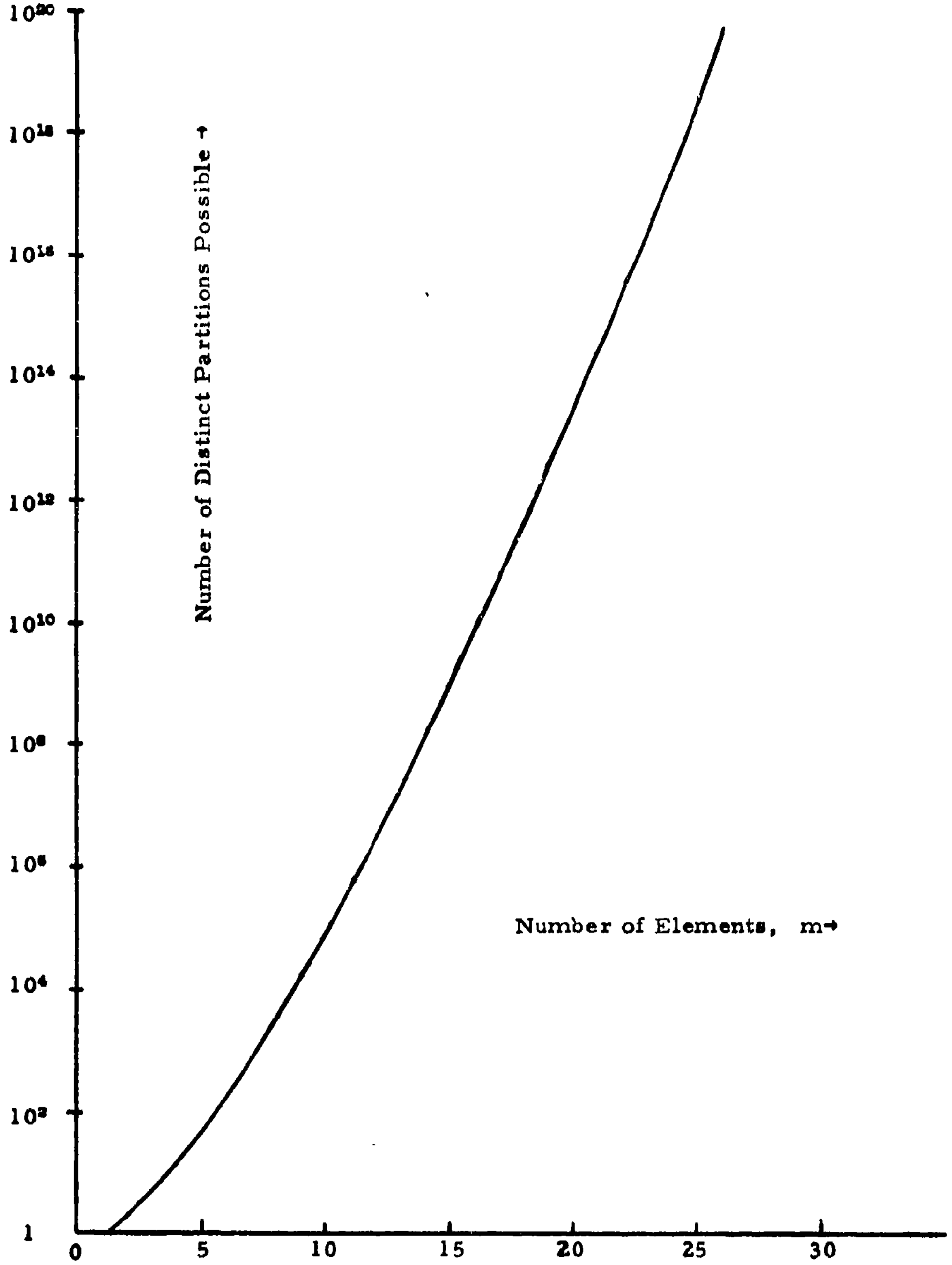


Figure 5. Growth of Number of Partitions Possible as $\mathfrak{U}$ Increases in Size

In the following discussion, it is useful to introduce a new parameter:

$$\lambda_{ij} = (\tau_{ij} + \tau_{ji})(1 - \frac{1}{2}\delta_{ij}). \tag{45}$$

It is the number of control transfers between α_i and α_j in either direction expected during a system epoch.

Dynamic Programming of Sequential Partitioning

Kernighan [18] has reported on an efficient method of sequential graph partitioning using dynamic programming techniques. The method could be applied to the problem of partitioning $\mathfrak{A}$ except for one major limitation. In Kernighan's formulation, each subset A_k' is completely defined by two integers i and j such that $1 \leq i \leq j \leq m$ and

$$A_k' = \{\alpha_k \mid k = i, i+1, \ldots, j\}. \tag{46}$$

The solution obtained is therefore generally not optimal. It is possible that the additional ordering constraint would be acceptable in some instances, in which case the method would be extremely valuable.

Integer Programming

As R contains redundant information, it is possible to define a vector $\underline{X}$ containing the $m(m-1)/2$ values appearing above the principal diagonal of R. As the diagonal elements of R are always equal to unity, R (and therefore the partition of $\mathfrak{A}$) can be constructed from the information contained in $\underline{X}$. If $\underline{L}$ is a vector containing elements τ_{ij} corresponding to the r_{ij} data positions in $\underline{X}$, then the optimal partition is one which maximizes

$$w = \underline{L} \cdot \underline{X}. \tag{47}$$

Constraints (37) and (39) need not be considered explicitly in this formulation. There are $(m/6)(m^2-3m+2)$ constraints of the form (43), and m constraints of the form (35). The latter may be placed in a conventional format if desired:

$$C\ \underline{X} \leq \underline{D}, \tag{48}$$

where C is a constant zero-one matrix of dimension $m \times m(m-1)/2$ which selectes certain terms of $\underline{X}$ and

$$\underline{D} = \begin{bmatrix} \mu_0 - g_1 \\ \mu_0 - g_2 \\ \vdots \\ \mu_0 - g_m \end{bmatrix} \quad . \tag{49}$$

Unit Merging

One form of this algorithm has been used for automatic program segmentation [6, 7, 8]. Coleman [17] describes it as a "greedy" algorithm; it seeks out the elements of $\mathfrak{A}$ corresponding to the larger values of λ_{ij} in a constantly updated penalty matrix. These elements are candidates for inclusion together in a subset A_k. Whenever candidates are included in the same subset, they are said to be "merged", and treated as if they were a single element in all further consideration.

Unit merging operates rapidly, and therefore can be used for relatively large systems. It is only an approximate method, however, and does not in general yield an optimal solution.

Backtrack Programming

This technique is an application of implicit enumeration methods [19, 20, 21]. In it, elements α_i are assigned to subsets A_k, one at a time. At the first step, α_1 is assigned. Following the second, α_1 and α_2 have been assigned. Following the mth step, all elements of $\mathfrak{A}$ have been assigned, since $|\mathfrak{A}| = m$, and so the complete partition of $\mathfrak{A}$ has been obtained. During this step-by-step partition creation process, a function y(i) gives the contribution of the first i elements of $\mathfrak{A}$ to Y. It is computed as follows:

$$y(0) = 1 , \tag{50}$$

and

$$y(i) = y(i-1) + \sum_{j=1}^{i-1} \lambda_{ij}(1-r_{ij}). \tag{51}$$

It can be demonstrated that y(m) = Y, and that y(i+1) $\geq$ y(i) for i = 1, 2, ... , m-1. This weakly monotonic nature of y(i) is important in the application of backtrack techniques to the synthesis of partitions which minimize Y. Consider the case in which $\alpha_1, \alpha_2, \ldots, \alpha_i$ have been assigned, and y(i) is found to be larger than some allowable upper limit[8]. Any partition of $\mathfrak{A}$ which includes identical[9] assignment of the first i elements of $\mathfrak{A}$ must also be unacceptable. Thus all such partitions are eliminated from further consideration.

A sophisticated algorithm for segmentation of computer code has been developed [6, 7, 8]; it uses backtrack methods to attack essentially the same problem as the present one. Absolute optimization is achieved by use of half-interval searching to set the threshold on Y, and the algorithm automatically saves partial results for use in subsequent trials. The computational load grows greatly as the number of control units increases, but the method is far superior to exhaustive searching.

PARTITIONS OF $\mathfrak{B}$

Determining an optimal partition of $\mathfrak{B}$ is clearly more complicated than finding such a partition of $\mathfrak{A}$. Although this is a present topic of investigation, it is useful to point out one particular subproblem which has been solved.

A function z(u) corresponding to y(i) can be derived:

$$z(0) = 0, \tag{52}$$

$$z(u) = z(u-1) + \sum_{j=1}^{m} q_{ju}[1 - \min(1, \sum_{v=1}^{u-1} q_{jv} s_{uv})]$$
$$\cdot [\gamma_j - \sum_{i=1}^{m} \lambda_{ij} \cdot \min(1, \sum_{v=1}^{u} q_{iv} s_{uv})]. \tag{53}$$

As required, z(n) = Z. However, certain stepwise optimization techniques require that the objective function be at least weakly monotonic, and this is not the case with z(u). Assuming that the β_u are assigned in an arbitrary order, Figure 6 shows by counterexample that there is no method of computing z(u) as a monotonic function of u, as the assignment of some β_u can actually reduce the number of boundary crossings. For convenience assume that none of the control units shown in Figure 6 is one from which control can enter or leave the system. Then $\gamma_1 = \tau_{12} = \gamma_2 = \tau_{23} = \gamma_3$. When β_1 is assigned to some B_k, there results $z(1) = \gamma_1 + \gamma_3 = 2\gamma_1$. If β_2 is assigned to the same B_k as β_1, the contribution resulting from this assignment is $\gamma_2 - \tau_{12} - \tau_{23} = -\gamma_1$, so $z(2) = \gamma_1$. This holds absolutely for any method of computing z(u), so for this structure z(2) < z(1).

This difficulty has been overcome by Peters, and the solution is described in another paper [16].

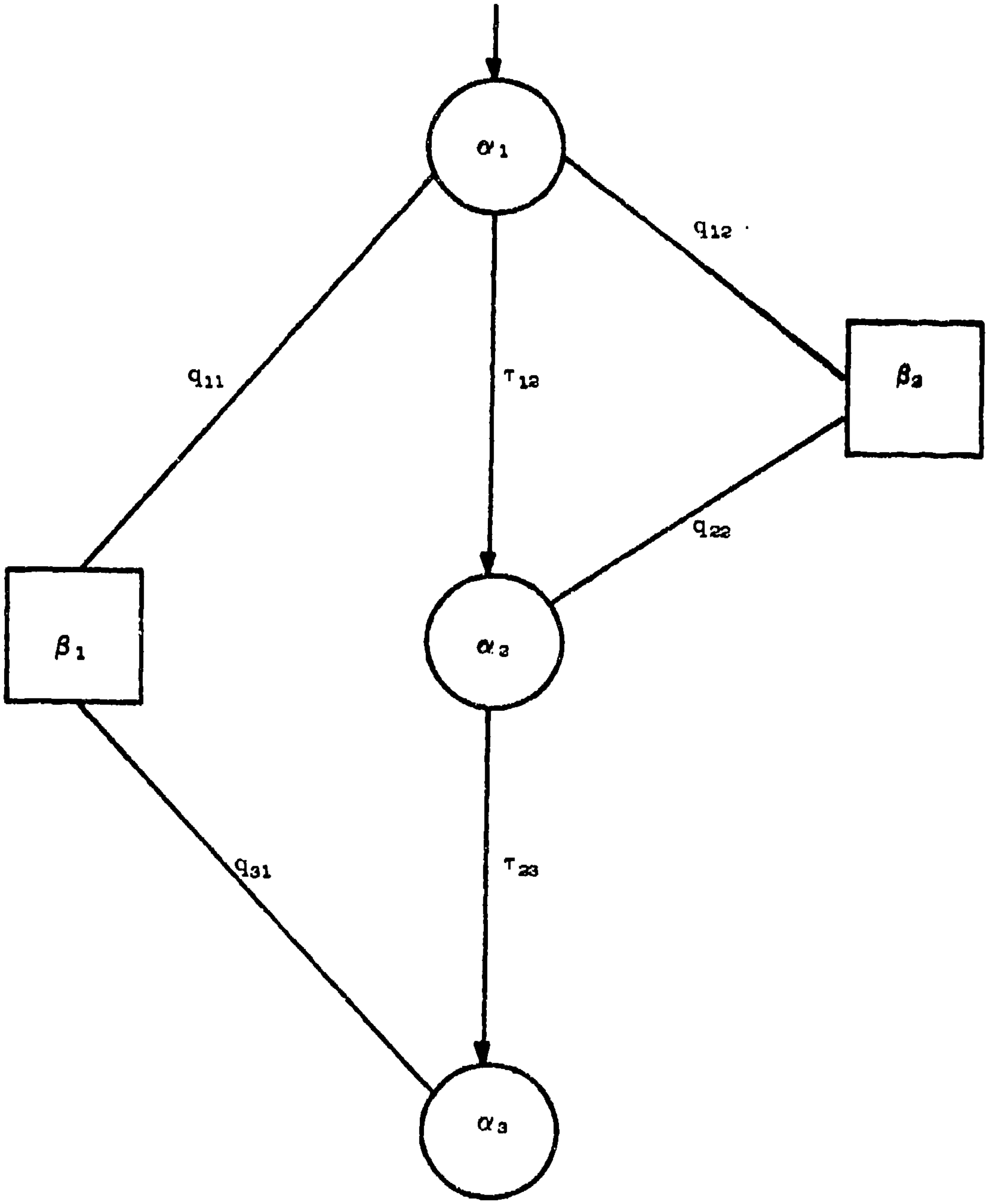


Figure 6. Non-monotonic Nature of z(i)

5. FURTHER WORK

This paper presents analysis of boundary crossing in a partitioned system model, and an introduction to the problem of synthesizing partitions in order to diminish the number of boundary crossings. The model may be applied either directly or with suitable modification, to the application areas mentioned in Section 1. In addition to the known structural parameters, required behavioral data may be obtained either from a priori knowlege of the probabilities $[p_{ij}]$ and $[e_i]$ involved, or from observation of $[\gamma_j]$ and $[\tau_{ij}]$. The cost of obtaining such data in one situation has been discussed [6]; the application of the model to any area of course requires that knowledge of the modelled system be available.

Significant topics remain for future investigation. These include classes of information systems in which several control units can be active at once, introducing the problem of simultaneous referencing of passive units by a multiplicity of control units. This necessitates consideration of contention among control units for passive units, the distinction between passive units of fixed and alterable content, and formation of access queues. Such problems arise in multi-user systems for information storage and retrieval, and data distribution. Multiprogrammed computer systems, in particular systems accessing mass storage devices, are candidates for such study.

In the present model, the penalty incurred for each boundary crossing is assumed to be constant, with respect to both the identification of the boundary and the state of the system. A more general model would be useful in connection with the study of information systems having a hierarchy of storage devices, and a variety of different costs for data transmission and control transfer.

Only a brief overview of the problem of partition synthesis is presented in this paper. Because of the magnitude of the computational problems involved, it is anticipated that synthesis methods will be developed for individual classes of application areas, and that only for small systems will the most general approaches be practical. One important consideration is the use of suboptimal solutions, weighing the cost of partition synthesis against the value returned in reduction of the number of boundary crossings. An example of this is the unit merging algorithm presented in Section 4. The use of peculiarities of the applications areas is another consideration. This has been emphasized by Golcomb [19] in connection with backtrack programming techniques; additional constraints may be introduced in order to reduce the feasible solution space, in come cases by orders of magnitude. This eliminates much of the computation required.

FOOTNOTES

The author is with Informatics Inc., Systems and Services Company, System Development Division, 6000 Executive Boulevard, Rockville, Maryland 20852. This work was supported by the Air Force Office of Scientific Research (AFSC), United States Air Force, under contract F44620-71-C-0070.

1. These parameters are γ_j and τ_{ij}, introduced in equations (9) and (11).

2. Q (and the matrices R and S, introduced below) are basically Boolean arrays. However, it is convenient in the development to use them in arithmetic expression. They are therefore designated to be real arrays, the elements of which are either zero or one.

3. In the following, explicit limits on subscripts are not given where they are clear from context; thus the condition "for k=1, 2, ... ,K" is understood in the statement that "$A_k \neq \Phi$".

4. For the present purposes, it is sufficient to restrict this to a chain having one ergodic set consisting of a single absorbing state.

5. Kral's S_{ij} is equivalent to this quantity [10].

6. The numerical values associated with some edges in that figure are used in a later example.

7. In the case that u=1, the summation from v=1 to v=u-1 is taken as zero.

8. Assignments which violate constraints of expressions (37) through (44) are not made.

9. Assignments are "identical" in this sense if they are isomorphic under a change of the subscripts of the sets A_k.

REFERENCES

[1] R.E. Nance, "An analytical model of a library network," J.Am.Soc.Inf.Sci., vol. 21, pp. 58-66, Jan.-Feb. 1970.

[2] Informatics Incorporated, "Automatic program segmentation," Rome Air Development Center, Griffiss AFB, N.Y. Rept. RADC-TR-67-634, Jan. 1968. (AD-827-720.)

[3] T.C. Lowe, "Analysis of Boolean models for time-shared, paged environments," C.Ass.Comput.Mach., vol. 12, pp. 199-205, Apr. 1969.

[4] ___________, "Automatic segmentation of cyclic program structures based on connectivity and processor timing," C.Ass.Comput.Mach., vol. 12, pp. 3-6, Jan. 1970.

[5] E.W. VerHoef, "Automatic program segmentation based on Boolean connectivity," in 1971 Spring Joing Comput. Conf., AFIPS Conf. Proc., vol. 38. Montvale, N.J.: AFIPS Press, 1971, pp. 491-495.

[6] T.C. Lowe, J.G.O. Van Dyke, and R.A. Colilla, "Program paging and operating algorithms," Rome Air Development Center, Griffiss AFB, N.Y. Rept. RADC-TR-68-444, Nov. 1968. (AD-845-758.)

[7] C.B. Peters, "Experiments in automatic paging," vol. 1, Rome Air Development Center, Griffiss AFB, N.Y. Rept. RADC-TR-71-231 (V.I.), Nov. 1971. (AD-734-253.)

[8] ___________, T.C. Lowe and A.J. Kreutzer, "Experiments in automatic paging," vol. 2, Rome Air Development Center, Griffiss AFB, N.Y. Rept. RADC-TR-71-231 (V.II), Nov. 1971. (AD-734-750.)

[9] B. Beizer, "Analytical techniques for the statistical evaluation of program running time," in 1971 Spring Joint Comput. Conf., AFIPS Conf. Proc., vol. 38. Montvale, N.J.: AFIPS Press, 1971, pp. 519-524.

[10] J. Kral, "One way of estimating frequencies of jumps in a program," C.Ass.Comput.Mach., vol. 11, pp. 475-480, July 1968.

[11] V.L. Wallace and R.S. Rosenberg, "Markovian models and numerical analysis of computer system behavior," in 1966 Spring Joint Comput. Conf., AFIPS Conf. Proc., vol. 28. Washington, D.C.: Spartan Books, 1966, pp. 141-148.

[12] C. V. Ramamoorthy, "Discrete Markov analysis of computer programs," in Assn. Comput. Mach. Proc. 20th Nat. Conf., vol. 20. New York: Lewis Winner, 1965, pp. 386-392.

[13] P. M. Morse, Library Effectiveness: A Systems Approach. Cambridge, Mass.: M.I.T. Press, 1968.

[14] W. Feller, An Introduction to Probability Theory and Its Applications. New York: John Wiley and Sons, 1950.

[15] J. G. Kemeny and J. L. Snell, Finite Markov Chains. Princeton, N. J.: D. Van Nostrand Co. Inc., 1960.

[1o] C. B. Peters, "Application of implicit enumeration to an information system assignment problem," submitted for publication.

[17] D. M. Coleman, "Finding quadratic groups--a quadratic programming approach in zero/one variables with applications," Ph. D. dissertation, University of Michigan, Ann Arbor, 1971.

[18] B. Kernighan, "Optimal sequential partitioning of graphs," J. Ass. Comput. Mach., vol. 18, pp. 34-40, Jan. 1971

[19] S. W. Golcomb and L. D. Baumbert, "Backtrack programming," J. Ass. Comput. Mach., vol. 4, pp. 516-624, Oct. 1965.

[20] R. L. Gue, J. C. Liggett, and K. C. Cain, "Analysis of algorithms for the zero-one programming problem," C. Ass. Comput. Mach., vol. 11, pp. 837-844, Dec. 1968.

[21] A. M. Geoffrion and R. E. Marsten, "Integer programming: a framework and state-of-the-art survey," Western Management Science Institute, University of California at Los Angeles Working Paper No. 172, May 1971. (PB-204-006.)